PTO 06-1114

Japanese Patent Application No. Sho 61[1986]-136145

CACHE MEMORY CONTROLLER CIRCUIT

Jun Hasegawa et. al.

UNITED STATES PATENT AND TRADEMARK OFFICE
WASHINGTON, D.C. DECEMBER 2005
TRANSLATED BY THE MCELROY TRANSLATION COMPANY

JAPANESE PATENT OFFICE
PATENT JOURNAL
KOKAI PATENT APPLICATION NO. SHO 61[1986]-136145

Int. Cl.[4]:	G 06 F 12/08
Sequence No. for Office Use:	8219-5B
Filing No.:	Sho 59[1984]-257589
Filing Date:	December 7, 1984
Publication Date:	June 24, 1986
No. of Invention:	1 (Total of 4 pages)
Examination Request:	Not requested

CACHE MEMORY CONTROLLER CIRCUIT
[Kyasshu memori seigyo kairo]

Inventors:	Jun Hasegawa et. al.
Applicants:	Hitachi, Ld. Hitachi Microcomputer Engineering K.K.

[There are no amendments to this patent.]

Claims

1. A cache memory controller circuit characterized in that in a data processor having a main memory and a cache memory, a means for storing whether or not a hit is found in the cache memory when accessing data is provided, whereby whether the data should be read directly from the main memory or they should be read from the main memory only if no hit is found when a search is run in the cache memory, is selected based on the content of said storage means when reading [the data].

2. The cache memory controller circuit described under Claim 1, characterized in that said memory means comprises 2 [storage means], namely, one for operand access and one for instruction retrieving.

Detailed explanation of the invention

Industrial application field

The present invention pertains to a cache memory controller circuit of a data processor equipped with a cache memory. In particular, it pertains to a cache memory controller circuit suitable for reducing an overhead in the event of a cache mishit.

Background of the invention

In the case of a convention data processor equipped with a cache memory, a search was first run in the cache memory when accessing data, and if the data searched were not found, the data was read from a main memory. When the data searched are not found in the cache memory, the time spent for running the search in the cache memory becomes an overhead. The proportion said overhead occupied in the average data access time was reduced by making the cache memory access time shorter than the main memory access time and by increasing the cache memory capacity so as to increase the likelihood that data searched were found in the cache memory. However, it is difficult to assure a sufficiently large cache memory capacity due to restrictions in terms of the level of integration when the cache is to be formed on-chip.

Although there is a method available (Kokai Patent Application No. Sho 55[1980]-9201) in which the hitting probability is improved by controlling the data to be held in the cache memory selectively, not much effect can be expected when the cache memory capacity is small.

Purpose of the invention

The purpose of the present invention is to present a cache controller circuit by which the influence of an overhead on the average data access time in the event of a mishit can be reduced even when the hitting probability is low due to a small cache memory capacity, and/or when the ratio between the cache memory search time and the main memory access time is not sufficiently high.

Outline of the invention

An overhead should be able to be reduced by starting to read the main memory as soon as the search in the cache memory begins when the possibility for an overhead is high. As far as retrieving an instruction by the data processor is concerned, a case in which same instruction group is executed repeatedly, and the size of instruction group is smaller than the capacity of the cache memory, may be mentioned as a case in which a hit is found in a cache memory with a small capacity. On the other hand, a case in which an instruction which has never been executed previously is newly executed and a case in which the size of the instruction group to be executed repeatedly is larger than the capacity of the cache memory may be mentioned as cases in which

no hit is found. As such, when a hit was found in the cache memory during the immediately previous instruction retrieving, it is highly likely that a hit is found in the cache memory during the next instruction retrieving; and when no hit was found in the cache memory during the immediately previous instruction retrieving, it is highly unlikely that a hit is found. Therefore, when the result of the previous retrieving instruction is stored, and whether a direct read from the main memory should be started or not is decided based on said [result], the overhead in the event of a mishit can be reduced.

An overhead in the event of a mishit can be eliminated if a search in the cache memory during the data access and the read from the main memory are always started simultaneously. However, because the time the processor occupies the bus to access the memory becomes long, the performance of the overall system drops. In addition, when the processor itself has finished processing data read from the cache memory and is about to begin the next data read, because the previous read from the main memory is not completed yet, it cannot start the read.

Application example of the invention

An application example of the present invention will be explained below using Figure 1.

Processor 1 is connected to cache memory 2 and memory unit 11 via address signal 100 and data signal 101. Cache memory 2 stores data and their addresses. Start signal 102 is a signal used to instruct to read data. Said signal is used also to instruct to read from cache memory 2. The data read from the cache memory are outputted to data signal 101 via tri-state buffer 3. Signal 104 is a signal which indicates whether or not data are present at an address indicated by address signal 100. Start signal 102 is connected to a strobe input on flip-flop 7 through delay element 9. Memory read signal 111 is an output signal from AND gate 8, and the signal used to instruct to output the result of read from an address in memory unit 11 as indicated by address 100 to data signal 101. Memory end signal 110 is a signal which is turned on when data are outputted from memory unit 11. End signal 103 is a signal used to notify processor 1 of the completion of data read, and processor 1 ends the operation of retrieving/reading the value of data signal 101 when it is turned on. Flip-flop 7 remembers whether a hit was found in the cache memory during the previous read or not.

First, a case in which a hit was found in the cache memory during the previous read, that is, 1 is stored in flip-flop 7, will be described.

Processor 1 outputs an address from which read should be executed to address signal 100 in order to turn start signal 102 on. Once start signal 102 becomes on, cache memory 2 begins to compare the addresses held therein with address signal 100. If a matching address is found, the stored data which are paired with the address are output to tri-state buffer 3, and signal 104 is turned on. If no matching address is found, signal 104 is turned off. Delay element 9 delays a

signal to the extent needed for running a search in cache memory 2. Therefore, signal 105 is turned on when the search in cache memory 2 is finished, and the result is indicated by signal 104. Once signal 105 becomes on, flip-flop 7 latches the value of signal 104. Delay element 10 is used to assure the delay of flip-flop 7, whereby it delays signal 105 until signal 106 is finalized after signal 105 is turned on and transfers it to signal 108 then.

When data applicable to the address are held in cache memory 2, 1 is latched newly in flip-flop 7, and signal 106 is turned on. Once signal 108 becomes on, signal 109 is turned on by AND gate 5, and tri-state buffer 3 outputs the data read from cache memory 2 to data signal 101. In addition, because signal 109 is connected to OR gate 4, end signal 103 is turned on, and the completion of read is notified to processor 1. In this case, processor 1 can obtain the data using [the same amount of] time as that spent for the read from cache memory 2.

When no data applicable to the address are held in cache memory 2, 0 is newly latched in flip-flop 7, and signal 106 is turned off. Thus, end signal 103 remains off even when signal 108 is turned on. On the other hand, signal 107 is turned on as signal 106 is turned off. Start signal 102 remains on until the read operation is completed. Thus, memory read signal 111 is turned on by AND gate 8, and an instruction is given to memory unit 11 to read data. Once the read is completed, the data are output to signal 101, and memory end signal 110 is turned on. Once memory end signal 110 becomes on, end signal 103 is turned on by OR gate 4, and the completion of the read is warned to processor 1. In this case, it takes the sum of the time spent for the read from cache memory 2 and the time spent for the read from memory unit 11 for processor 1 to obtain the data.

Next, the case will be described when no hit is found in the cache memory during the previous read, that is, when o is stored in flip-flop 7.

Processor 1 outputs an address, from which read should be executed, to address signal 100 in order to turn start signal 102 on. Once start signal 102 becomes on, because 0 is held in flip-flop 7, and signal 107 is on, memory read signal 111 is turned on immediately by AND gate 8 in order to send a read instruction to memory unit 11. On the other hand, a search in and a read from the cache memory are executed in the same way as those executed when a hit is found.

When data applicable to the address are held in cache memory 2, as described above, AND gate 5 turns signal 109 on, and the data in cache memory 2 are output to data signal 101 via tri-state buffer 3, and end signal 103 is turned on also. Therefore, processor 1 can obtain the data using the same amount of time as that spent for the read from cache memory 2.

When no data applicable to the address are held in cache memory 2, end signal 103 is turned on as memory end signal 110 begins to be turned on. Therefore, processor 1 can obtain the data using the same amount of time as that spent for reading memory unit 11.

As described above, in the present application example, data can be obtained using the cache memory read time when a hit is found in the cache memory, or using the memory unit read time even when no hit is found if no hit was found during the previous read.

Although only one flip-flop, which held the previous cache search result, was provided in the aforementioned application example, when said [flip-flop] is divided into two, namely, one for instruction retrieving and another for data accessing, the probability for accurate hit prediction can be improved. In addition, when the data accessing flip-flop is provided for each access address generation method (absolute address access, indexed [access], etc.), the prediction hitting rate can be improved.

Effect of the invention

According to the present invention, when no hit was found during the previous data access, access to the main memory is initiated immediately during the next data access without waiting for the result of a search run in the cache memory, so that an overhead can be reduced because the hitting probability during the next access changes depending on the immediately preceding search result as is evident in the retrieving of an instruction.

Brief description of the figure

Figure 1 is a block diagram of a processor with a built-in cache to which the present invention is applied.

1 ... processor's main body; 2 ... cache memory; 3 ... tri-state buffer; 7 ... flip-flop; 9, 10 ... delay element; 100 ... address signal; 101 ... data signal; 102 ... start signal; 103 ... end signal; 111 ... memory read signal; 110 ... memory end signal; and 11 ... memory unit.

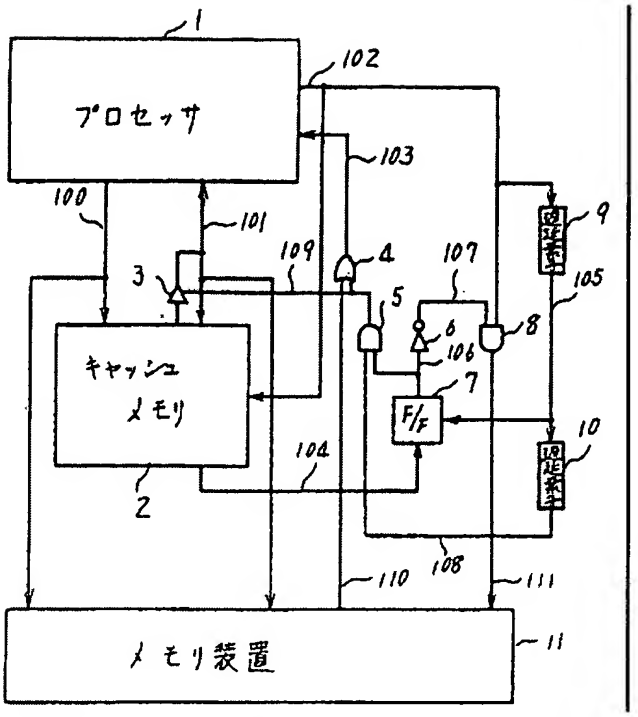

Figure 1

Key:
1 Processor
2 Cache memory
9, 10 Delay element
11 Memory unit